Amendments to the Claims:

Please amend claims 1, 5, 6, 8, 9,10, 13, 17, 18, 19 22, 25, 33, 34, 37, 41, 42, 43, 44, 45, 46 as indicated in the listing of claims, below. This listing of claims is a complete set of pending claims and will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (currently amended) An apparatus comprising:

a key generator to generate an operating system nub key (OSNK) unique to an operating system (OS) nub, the OS nub being part of an operating system to run ~~running~~ on a secure platform, the OS nub further being associated with a ring hierarchy level; and

a usage protector coupled to the key generator to protect usage of a subset of a software environment using the OSNK.

2. (original) The apparatus of claim 1 wherein the key generator comprises:

a combiner to combine an identification of the OS nub and a master binding key (BK0) of the secure platform, the combined identification and the BK0 corresponding to the OSNK.

3. (original) The apparatus of claim 2 wherein the identification is a hash value of one of the OS nub and a certificate representing the OS nub.

4. (original) The apparatus of claim 1 wherein the usage protector comprises:

an encryptor to encrypt the subset of the software environment using the OSNK.

5. (currently amended) The apparatus of claim 1 wherein the usage protector comprises:

a first encryptor to encrypt a first hash value of the subset of the software environment using the OSNK[,] and to store the encrypted first hash value ~~being stored~~ in a storage;

a second encryptor to ~~encrypted~~ encrypt a second hash value of the subset of the software environment using the OSNK; and

a comparator to compare result between the encrypted second hash value and the encrypted first hash value retrieved from the storage, the comparison result indicating if the subset of the software environment has been modified.

6. (currently amended) The apparatus of claim 1 wherein the usage protector comprises:

a first encryptor to encrypt a first hash value of the subset of the software environment using the OSNK[,] and to store the encrypted first hash value ~~being stored~~ in a storage;

a decryptor to decrypt the encrypted first hash value using the OSNK; and

a comparator to compare ~~result between~~ a second hash value and the decrypted first hash value retrieved from the storage, to determine ~~the comparison result indicating~~ if the subset of the software environment has been modified.

7. (original) The apparatus of claim 1 wherein the usage protector comprises:

a signature generator to generate a signature of the subset of the software environment using a private key; and

a signature verifier to verify the signature using a public key to protect usage of subset if the subset of the software environment has been modified.

8. (currently amended) The apparatus of claim 1 wherein the usage protector comprises:

a manifest generator to generate a manifest of the subset of the software environment, the manifest ~~describing~~ to describe the ~~potion~~ portion of the software environment;

a signature generator coupled to the manifest generator to generate a manifest signature of the manifest using a private key~~;~~

a decryptor to decrypt the private key ~~being decrypted by a decryptor~~ using the OSNK;

a signature verifier coupled to the signature generator to verify the manifest signature using a public key; and

a manifest verifier coupled to the signature verifier to verify the manifest, the verified manifest to indicate ~~indicating~~ if the subset of the software environment has been modified.

9. (currently amended) The apparatus of claim 1 wherein the secure platform ~~uses~~ is to use an isolated execution mode.

10. (currently amended) The apparatus of claim 1 wherein the software environment is one of a ~~Windows~~ WINDOWS® operating system, a ~~Windows~~ WINDOWS® 95 operating system, a ~~Windows~~ WINDOWS® 98 operating system, a ~~Windows~~ WINDOWS NT® operating system, and a ~~Windows~~ WINDOWS® 2000 operating system.

11. (original) The apparatus of claim 9 wherein the subset of the software environment is a registry of an operating system.

12. (original) The apparatus of claim 2 wherein the BK0 is generated at random on a first invocation of a processor nub.

13. (currently amended) A method comprising:

generating an operating system nub key (OSNK) unique to an operating system (OS) nub, the OS nub being part of an operating system ~~running~~ to run on a secure platform, the OS nub further being associated with a ring hierarchy level; and

protecting usage of a subset of the software environment using the OSNK.

14. (original) The method of claim 11 wherein generating the OSNK comprises:

combining an identification of the OS nub and a master binding key (BK0) of the secure platform, the combined identification and the BK0 corresponding to the OSNK.

15. (original) The method of claim 12 wherein the identification is a hash value of one of the OS nub and a certificate representing the OS nub.

16. (original) The method of claim 11 wherein protecting usage comprises:

encrypting the subset of the software environment using the OSNK.

17. (currently amended) The method of claim 11 wherein protecting usage comprises:

encrypting a first hash value of the subset of the software environment by the OSNK[,] and storing the encrypted first hash value ~~being stored~~ in a storage;

encrypting a second hash value of the subset of the software environment using the OSNK ~~decrypting the encrypted first hash value of the subset of the software~~; and

comparing the encrypted second hash value to the encrypted first hash value retrieved from the storage, the comparison result indicating if the subset of the software environment has been modified.

18. (currently amended) The method of claim 11 wherein protecting usage comprises:

encrypting a first hash value of the subset of the software environment by the OSNK[,] and storing the encrypted first hash value ~~being stored~~ in a storage;

decrypting the encrypted first hash value using the OSNK; and

comparing a second hash value to the decrypted first hash value retrieved from the storage, ~~the comparison result indicating~~ to determine if the subset of the software environment has been modified.

19.[.] (currently amended) The method of claim 11 wherein protecting usage comprises:

generating a signature of the subset of the software environment using a private key; and

verifying the signature using a public key to detect if the subset of the software environment has been modified.

20. (original) The method of claim 11 wherein detecting comprises:

generating a manifest of the subset of the software environment, the manifest describing the subset of the software environment;

generating a manifest signature of the manifest using a private key, the private key being decrypted using the OSNK;

verifying the encrypted manifest signature using a public key; and

verifying the manifest, the verified manifest indicating if the subset of the software environment has been modified.

21. (original) The method of claim 11 wherein the secure platform uses an isolated execution mode.

22. (currently amended) The method of claim 11 wherein the software environment is one of a ~~Windows~~ WINDOWS® operating system, a ~~Windows~~ WINDOWS® 95 operating system, a ~~Windows~~ WINDOWS® 98 operating system, a ~~Windows~~ WINDOWS NT® operating system, and a ~~Windows~~ WINDOWS® 2000 operating system.

23. (original) The method of claim 19 wherein the subset of the software environment is a registry of the operating system.

24. (original) The method of claim 14, wherein the BK0 is generated at random on a first invocation of a processor nub.

25. (currently amended) A computer program product comprising:

a computer usable medium having computer program code embodied therein, the computer program product having:

computer readable program code for generating an operating system nub key (OSNK) unique to an operating system (OS) nub, the OS nub being part of an operating system ~~running~~ to run on a secure platform, the OS nub further being associated with a ring hierarchy level; and

computer readable program code for protecting usage a subset of the software environment using the OSNK.

26. (original) The computer program product of claim 21 wherein the computer readable program code for generating the OSNK comprises:

computer readable program code for combining an identification of the OS nub and a master binding key (BK0) of the secure platform, the combined identification and the BK0 corresponding to the OSNK.

27. (original) The computer program product of claim 22 wherein the identification is a hash value of one of the OS nub and a certificate representing the OS nub.

28. (original) The computer program product of claim 21 wherein the computer readable program code for protecting usage comprises:

computer readable program code for encrypting the subset of the software environment using the OSNK.

29. (original) The computer program product of claim 21 wherein the computer readable program code for protecting usage comprises:

computer readable program code for encrypting a first hash value of the subset of the software environment using the OSNK, the encrypted first hash value being stored in a storage;

computer readable program code for encrypting a second hash value of the subset of the software environment using the OSNK; and

computer readable program code for comparing the encrypted second hash value and the encrypted first hash value retrieved from the storage, the comparison result indicating if the subset of the software environment has been modified.

30. (original) The computer program product of claim 21 wherein the computer readable program code for protecting usage comprises:

computer readable program code for encrypting a first hash value of the subset of the software environment using the OSNK, the encrypted first hash value being stored in a storage;

computer readable program code for decrypting the encrypted first hash value of the subset of the software environment using the OSNK; and

computer readable program code for comparing the second hash value and the decrypted first hash value retrieved from the storage, the comparison result indicating if the subset of the software environment has been modified.

31. (original) The computer program product of claim 21 wherein the computer readable program code for protecting usage comprises:

computer readable program code for generating a signature of the subset of the software environment using a private key, the private key being decrypted using the OSNK; and

computer readable program code for verifying the signature using a public key to detect if the subset of the software environment has been modified.

32. (original) The computer program product of claim 21 wherein the computer readable program code for protecting usage comprises:

computer readable program code for generating a manifest of the subset of the software environment, the manifest to describe ~~describing~~ the subset of the software environment;

computer readable program code for generating a manifest signature of the manifest using a private key that has been ~~, the private key being~~ decrypted using the OSNK;

computer readable program code for verifying the manifest signature using a public key; and

computer readable program code for verifying the manifest, the verified manifest indicating if the subset of the software environment has been modified.

33. (currently amended) The computer program product of claim 21 wherein the secure platform ~~uses~~ is to use an isolated execution mode.

34. (currently amended) The computer program product of claim 21 wherein the software environment is one of a ~~Windows~~WINDOWS® operating system, a ~~Windows~~ WINDOWS® 95 operating system, a ~~Windows~~ WINDOWS® 98 operating system, a ~~Windows~~ WINDOWS NT® operating system, and a ~~Windows~~ WINDOWS® 2000 operating system.

35. (original) The computer program product of claim 29 wherein the subset of the software environment is a registry of an operating system.

36. (original) The computer program product of claim 26 wherein the BK0 is generated at random on a first invocation of a processor nub.

37. (currently amended) A system comprising:

a processor;

a storage coupled to the processor, the storage storing a subset of a software environment; and

a usage protector comprising:

a key generator to generate an operating system nub key (OSNK) unique to an operating system (OS) nub, the operating system nub being part of a software environment ~~running~~ to run on a secure platform, the operating system nub further being associated with a ring hierarchy level; and

a usage protector coupled to the key generator to detect a subset of the software environment using the OSNK.

38. (original) The system of claim 31 wherein the key generator comprises:

a combiner to combine an identification of the operating system nub and a master binding key (BK0) of the secure platform, the combined identification and BK0 corresponding to the OSNK.

39. (original) The system of claim 32 wherein the identification is a hash value of one of the OS nub and a certificate representing the OS nub.

40. (original) The system of claim 31 wherein the usage protector comprises:

an encryptor to encrypt the subset of the software environment using the OSNK.

41. (currently amended) The system of claim 31 wherein the usage protector comprises:

an encryptor to encrypt a first hash value of the subset of the software environment using the OSNK[,] and to provide the encrypted first hash value ~~being stored in~~ to a storage; and

a comparator to compare the encrypted second hash value and the encrypted first hash value retrieved from the storage~~, the comparison result indicating~~ to determine if the subset of the software environment has been modified.

42. (currently amended) The system of claim 31 wherein the usage protector comprises:

an encryptor to encrypt a first hash value of the subset of the software environment using the OSNK[,] and to store the encrypted first hash value ~~being stored~~ in a storage; and

a comparator to compare the second hash value and the decrypted first hash value retrieved from the storage[,] to determine ~~the comparison result indicating~~ if the subset of the software environment has been modified.

43. (currently amended) The system of claim 31 wherein the usage protector comprises:

a signature generator to generate a signature of the subset of the software environment using a private key~~, the private key being decrypted~~ and to decrypt the private key using the OSNK; and

a signature verifier to verify the encrypted signature using a public key to detect if the subset of the software environment has been modified.

44. (currently amended) The system of claim 31 wherein the usage protector comprises:

a manifest generator to generate a manifest of the subset of the OS, the manifest ~~describing~~ to describe the ~~potion~~ portion of the software environment;

a signature generator coupled to the manifest generator to generate a manifest signature of the manifest using a private key[,] and to decrypt the private key ~~being decrypted~~ using the OSNK;

a signature verifier coupled to the encryptor to verify the encrypted manifest signature using a public key; and a manifest verifier coupled to the signature verifier to verify the manifest~~[,] the verified manifest indicating~~ to indicate if the subset of the software environment has been modified.

45. (currently amended) The system of claim 31 wherein the secure platform ~~uses~~ is to use an isolated execution mode.

46. (currently amended) The system of claim 31 wherein the software environment is one of a ~~Windows~~ WINDOWS® operating system, a ~~Windows~~WINDOWS® 95 operating system, a ~~Windows~~ WINDOWS® 98 operating system, a ~~Windows~~ WINDOWS NT® operating system, and a ~~Windows~~ WINDOWS® 2000 operating system.

47. (original) The system of claim 39 wherein the subset of the software environment is a registry of an operating system.

48. (original) The system of claim 38 wherein the BK0 is generated at random on a first invocation of a processor nub.